BSM622

Veri Analitiği ve İstatistik

6. Hafta: Sınıflandırma

Doc. Dr. Onur Polat

Kaynak: Bruce, Bruce, Gedeck (2026). AI-Assisted Statistics for Data Scientists, 3. Baski, O'Reilly - Bolum 5

Bu Haftanin Icerigi

- Sınıflandırma problemi: İkili ve çok sınıflı
- Naive Bayes sınıflandırıcı
- Diskriminant Analizi (LDA)
- Lojistik Regresyon (GLM, logit, odds oranları)
- Sınıflandırma Modellerinin Değerlendirilmesi
- Confusion Matrix, Precision, Recall, Specificity
- ROC Eğrisi, AUC ve Lift
- Dengesiz Veri İçin Stratejiler (SMOTE, ağırlıklandırma, maliyet)

Siniflandirma Nedir?

- Hedef değişken Y kategoriktir (sınıf etiketi)
- İkili sınıflandırma: Y = 1 (ilgi sınıfı) veya Y = 0
- Örnekler: Kredi temerrüüu (evet/hayır), müşteri terk, phishing e-posta
- Çok sınıflı: Gmail'in gelen kutusunu "birincil / sosyal / tanıtım / forum" olarak sınıflandırması
- Denetimli öğrenme: Etiketli eğitim verisi üzerinde model eğitilir, bilinmeyen etiketli yeni veriye uygulanır
- Çoğu algoritma hem predict (sınıf) hem predict_proba (olasılık skoru) üretir
- Kesim değeri (cutoff): eşikten yüksek olasılıklar ilgi sınıfına atanır (varsayılan 0.5)

Sınıflandırma - Anahtar Terimler

- Eğilim skoru (propensity score): Bir gözlemin ilgi sınıfına ait olma olasılığı
- Kesim (cutoff): Olasılığı sınıf etiketine çeviren eşik değeri
- Karışıklık Matrisi: Doğru ve yanlış sınıflandırmaları gösteren tablo
- Precision: Tahmin edilen pozitiflerin ne kadarının gerçekten pozitif olduğu
- Recall (duyarlılık): Gerçek pozitiflerin ne kadarının yakalandığı
- Specificity: Gerçek negatiflerin ne kadarının yakalandığı
- AUC: ROC eğrisi altındaki alan – eşikten bağımsız model kalitesi

BÖLÜM I

Naive Bayes

Bayesyen Siniflandirma

- Amaç: X1,...,Xp değerleri gözlendiğinde Y=i olması olasılığı P(Y=i|X1,...,Xp)
- Tam Bayesyen çözüm: Aynı X değerlerine sahip tüm kayıtları bul, en yaygın sınıfı ata
- Problem: Değişken sayısı arttıkça TAM eşleşme bulmak imkânsızlaşır
- Örneğin 8 değişkenli siyasi tercih tahmininde tek bir eşleşme bile olmayabilir
- Naive varsayım: Y verildiğinde özellikler koşullu bağımsız
- Bu basitleştirme ile her özellik için P(Xj|Y=i) ayrı ayrı tahmin edilir
- Dikkat: Naive Bayes KATEGORİK (faktor) öncüller gerektirir

Naive Bayes Formulu

$$P(Y=i \mid X1,X2,...,Xp) \text{ prop. } P(Y=i) \text{ prod_j } P(Xj \mid Y=i)$$

- Her sınıf i için: Prior olasılık P(Y=i) ve her özelliğin koşullu olasılığı P(Xj|Y=i) çarpılır
- Normalizasyon: Tüm sınıfların toplamı 1 olacak şekilde bölünür
- En yüksek olasılıklı sınıf atanır
- Hesaplanan posterior olasılığı tahminler yanlı (biased) olabilir
- Ancak kayıtları P(Y=1) olasılığına göre sıralamak önemli ise Naive Bayes iyi sonuç verir
- Laplace yumuşatma: Eğitimde görülmeyen kategorilere sıfır olasılık atanmasını önler

Python ile Naive Bayes (loan_data)

```
import pandas as pd
from sklearn.naive_bayes import MultinomialNB

loan_data = pd.read_csv('loan_data.csv.gz')

predictors = ['purpose_', 'home_', 'emp_len_']
outcome = 'outcome'

X = pd.get_dummies(loan_data[predictors],
                   prefix='', prefix_sep='',
                   drop_first=True, dtype=int)
y = loan_data[outcome]

naive_model = MultinomialNB(alpha=0.01, fit_prior=True)
naive_model.fit(X, y)

# Yeni bir gozlem icin tahmin
print('Sinif:', naive_model.predict(X.iloc[146:147])[0])
print('Olasiliklar:', naive_model.predict_proba(X.iloc[146:147])[0])
```

Sayısal Öngörüler ile Naive Bayes

- Standart Naive Bayes KATEGORİK öngörücü gerektirir
- Sayısal değişkenler için iki yaklaşım vardır:
- 1) Olasılık modeli kullan: Normal dağılım ile P(Xj|Y=i) tahmin et
- 2) Sayısal öncüyü BIN'lere ayırıp kategorikleştir
- scikit-learn'de GaussianNB sayısal değişkenler için, MultinomialNB kategorikler için uygundur
- Karma veri için: ColumnTransformer + OneHotEncoder + KBinsDiscretizer ile pipeline kurulabilir
- Örnek: loan_data'da borrower_score sayısal, purpose_ kategorik - her ikisini de kullanmak için KBinsDiscretizer gerekir

BÖLÜM II

Diskriminant Analizi (LDA)

LDA - Linear Discriminant Analysis

- Fisher (1936) tarafından önerilen EN ESKİ istatistiksel sınıflandırıcı
- Ortak noktası: Verileri sınıflar arasındaki ayrımı maksimize edecek doğrusal kombinasyonlar bulur
- Varsayım: Öngörüler sınıflar içinde normal dağılım gösterir. LDA her iki sınıf için AYNI kovaryans matrisini varsayar.
- QDA (Quadratic): Her sınıf icin FARKLI kovaryans matrisi (daha esnek)
- Modern sınıflandırmada ağaç modelleri ve lojistik regresyon LDA'nın yerini aldı
- Eğitim hızlı, yorumlanabilir, PCA ile bağlantılı

Fisher'in Dogrusal Diskriminanti

- Amaç: SSbetween / SSwithin oranını maksimize eden doğrusal kombinasyonu bulmak
- SSbetween: Sınıflar arası kareler toplamı (grup ortalama farkının karesi)
- SSwithin: Sınıflar içi kareler toplamı (grup içi varyans)
- Kovaryans matrisi: Her özellik çiftinin birlikte nasıl değiştiğini ölçer
- Mahalanobis uzaklığı: Standart sapmayı çoklu değişkenlere genelleme uzaklık ölçümü

maksimize et: SS_between / SS_within

Python ile LDA (loan3000 veri seti)

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

loan3000 = pd.read_csv('loan3000.csv')
loan3000.outcome = loan3000.outcome.astype('category')

predictors = ['borrower_score', 'payment_inc_ratio']
outcome = 'outcome'

X = loan3000[predictors]
y = loan3000[outcome]

loan_lda = LinearDiscriminantAnalysis()
loan_lda.fit(X, y)

print('Diskriminant agirliklari:')
print(pd.DataFrame(loan_lda.scalings_, index=X.columns))

# Olasilik tahminleri
pred = pd.DataFrame(loan_lda.predict_proba(X),
                    columns=loan_lda.classes_)
print(pred.head())
```

BOLUM III

Lojistik Regresyon

Logit ve Lojistik Yanıt Fonksiyonu

- İkili yanıtlı regresyonda temel problem: Doğrusal model (0,1) aralığı dışında tahmin verir
- Çözüm: Logit dönüşümü ile olasılığı (-inf, +inf) aralığına taşıyoruz
- Odds (bahis oranı): P / (1-P) - "başarı" sayısının "başarısızlık" sayısına oranı
- Log-odds (logit): log(P / (1-P)) - artik dogrusal modellenebilir
- Sigmoid fonksiyonu logit'in tersidir: olasılığı (0,1) aralığına geri çevirir

$$\log(P / (1-P)) = b0 + b1*X1 + \ldots + bq*Xq$$

$$p = 1 / (1 + \exp(-(b0 + b1*X1 + \ldots + bq*Xq)))$$

Python ile Lojistik Regresyon

```
from sklearn.linear_model import LogisticRegression

loan_data = pd.read_csv('loan_data.csv.gz')

predictors = ['payment_inc_ratio', 'purpose_', 'home_',
              'emp_len_', 'borrower_score']
outcome = 'outcome'

X = pd.get_dummies(loan_data[predictors],
                   prefix='', prefix_sep='',
                   drop_first=True, dtype=int)
y = loan_data[outcome]

# C=np.inf -> regularizasyon kapali (saf MLE)
import numpy as np
logit_reg = LogisticRegression(C=np.inf, max_iter=1000)
logit_reg.fit(X, y)

print('Katsayilar:', dict(zip(X.columns, logit_reg.coef_[0])))
print('Siniflar:', logit_reg.classes_)
```

Genellestirilmis Dogrusal Modeller (GLM)

- GLM iki temel bileşenden oluşur:
- 1) Olasılık dağılımı (family): Binomial (lojistik), Poisson (sayma), Gamma, vb.
- 2) Link fonksiyonu: Yanıt değişkenini öngörücülere bağlayan dönüşüm (logit, log, identity, probit...)
- Lojistik regresyon = Binomial family + logit link
- Doğrusal regresyon = Gaussian family + identity link
- MLE (Maximum Likelihood Estimation): Kapalı form yok; iteratif optimizasyon (Fisher scoring)
- Deviance: Model kalite ölçütü - log-likelihood'un -2 katı

Katsayilarin Yorumlanmasi - Odds Oranlari

- Doğrusal regresyonda bj = Xj'de 1 birim artış -> Y'de bj birim değişim Lojistik regresyonda: bj = Xj'de 1 birim artış -> log(odds)'ta bj birim degisim
- Pratik yorumlama: exp(bj) ODDS ORANIDIR
- Ornek (loan_data): purpose_small_business katsayisi = 1.21526
- -> exp(1.21526) = 3.4 -> Küçük işletme kredisi, kredi kartı borcu kapatmaya göre 3.4 KAT daha riskli
- borrower_score katsayısı = -4.6 -> en iyi borçluların temerrüt olasılığı 100 kat daha düşük
- Lineer olmayan ilişki: Katsayılar logaritmik ölçekte doğrusal, olasılık ölçeğinde S-eğrisi

statsmodels ile Detayli Analiz

```
import statsmodels.api as sm

y_numbers = [1 if yi == 'default' else 0 for yi in y]
logit_reg_sm = sm.GLM(y_numbers,
                      X.assign(const=1),
                      family=sm.families.Binomial())
logit_result = logit_reg_sm.fit()
print(logit_result.summary())

# Cikti:
# - Her katsayi icin Std.Error, z-value, p-value
# - Null deviance, Residual deviance, AIC
# - Fisher Scoring iterasyon sayisi

# Odds oranlari ve guven araliklari
import numpy as np
odds = pd.DataFrame({
    'coef': logit_result.params,
    'odds_ratio': np.exp(logit_result.params),
    'p_value': logit_result.pvalues
}).round(4)
print(odds)
```

BOLUM IV

Model Değerlendirme

Dogruluk (Accuracy) ve Sinirlari

- Accuracy = (TP + TN) / Toplam gozlem sayisi
- En basit ve en sık kullanılan metrik
- ÖNEMLİ SORUN: Dengesiz veri de son derece yanıltıcı!
- Örnek: %99 sağlıklı / %1 hasta olduğu bir hastalık taramasında herkese "sağlıklı" diyen model %99 accuracy alır ama HİÇBİR hastayı tespit edemez
- Finansal dolandırıcılık: %0.1 oranla dolandırıcı veride %99.9 accuracy ile model useless olabilir
- Cozum: Precision, Recall, F1, AUC gibi daha nüanslı metrikler kullanmak

Confusion Matrix (Karışıklık Matrisi)

- Dogru ve yanlis siniflandirmalari TURE GORE gosteren 2x2 tablo (ikili sınıflandırmada)

	Tahmin: 1	Tahmin: 0
Gercek: 1	TP (Dogru Pozitif)	FN (Yanlis Negatif)
Gercek: 0	FP (Yanlis Pozitif)	TN (Dogru Negatif)

Note: İlgi sınıfı genellikle Y=1 olarak etiketlenir (temerrüt, dolandırıcılık, hastalık)

Precision, Recall, Specificity

- Precision = TP / (TP + FP) -> "Pozitif dediğimde ne kadar haklıyım?"
- Recall (Sensitivity) = TP / (TP + FN) -> " Gerçek pozitiflerin yüzde kaçını yakaladım?"
- Specificity = TN / (TN + FP) -> «Gerçek negatiflerin yuzde kacini yakaladim?"
- F1 Skoru = 2 * (P * R) / (P + R) -> Precision ve Recall'un harmonik ortalamasi
- İşletme etkisi: Kredi kartı sahtekârlığı tespiti -> Yüksek Recall önemli (kaybı en aza indir)
- Pazarlama kampanyası -> Yüksek Precision önemli (bütçeyi boşa harcama)
- Tibbi tarama -> Her ikisini dengele; false negatif = kaçırılan hasta, false pozitif = gereksiz test

Nadir Sınıf Problemi

- Birçok uygulamada ilgi sınıfı NADİR'dir: dolandırıcılık, temerrüt, hastalık, satın alma
- Nadir sınıf (1) daha önemlidir - genellikle yanlış sınıflandırmanın maliyeti yüksektir.
- Örnek: %0,1'lik web ziyaretçisi satın alırsa, "herkes satın almayacak" diyen model %99,9 doğruluk alır.
- Ama bu model gerçek satın alanları tespit etmiyor
- Çözüm stratejileri (sonra detaylı): Resampling, ağırlıklandırma, maliyete dayalı sınıflandırma, eşik ayarlama
- Değerlendirme: Accuracy yerine Precision, Recall, F1, AUC kullanılmalı

ROC Egrisi ve AUC

- ROC = Receiver Operating Characteristic (Alıcı İşletim Karakteristiği)
- Eşikten BAĞIMSIZ model kalitesi değerlendirme aracı
- Y ekseni: Recall (TPR, True Positive Rate)
- X ekseni: 1 - Specificity (FPR, False Positive Rate)
- Kesim değerini 0 -> 1 arasında tararken Recall vs. FPR grafiği çizilir
- AUC = ROC egrisi altindaki alan (0.5 = rastgele, 1.0 = mukemmel)
- AUC 0.5-0.6 kötü, 0.7-0.8 iyi, 0.9+ mükemmel kabul edilir
- Dengesiz veride Precision-Recall eğrisi daha bilgilendiricidir

Python ile ROC ve AUC Hesaplama

```
from sklearn.metrics import (roc_curve, roc_auc_score,
                             confusion_matrix,
                             classification_report)
import matplotlib.pyplot as plt
# Olasilik tahminleri
y_proba = logit_reg.predict_proba(X)[:, 0]  # 'default' sinifi

# ROC egrisi
fpr, tpr, thresholds = roc_curve(
    [1 if yi=='default' else 0 for yi in y], y_proba)
auc = roc_auc_score(
    [1 if yi=='default' else 0 for yi in y], y_proba)

plt.figure(figsize=(6,6))
plt.plot(fpr, tpr, label=f'AUC = {auc:.3f}')
plt.plot([0,1], [0,1], 'k--', alpha=0.5)
plt.xlabel('False Positive Rate (1 - Specificity)')
plt.ylabel('True Positive Rate (Recall)')
plt.title('ROC Egrisi - Lojistik Regresyon (loan_data)')
plt.legend()
plt.grid(alpha=0.3)
plt.show()
```

Lift (Yükselim) ve Gains Chart

- Problem: Nadir sınıfta kesim < 0.5 yapmak gerekebilir, ama hangi değere?
- Lift bu sorunun cevabına yardımcı olur
- Kayıtları tahmin olasılığına göre sıralayıp decile'lara böl
- En yüksek decile'daki pozitif oranını genel pozitif oranına böl -> Lift
- Örnek: Gerçek pozitif oranı %0.1, ancak en yüksek decile'da %0.3 -> Lift = 3
- Pazarlama uygulamasında çok yaygın: Doğrudan posta kampanyaları, mükemmel aday sıralaması
- Uplift (artis): A/B testi + model birleşimi; her bir birey için tedavi A vs B'nin BİREYSEL etkisi

BÖLÜM V

Dengesiz Veri Icin Stratejiler

Undersampling (Alt Orneklem)

- Problem: loan_data'da %19 temerrüt, %81 ödeme -> model ödemeleri ezberliyor
- Tam veriyle eğitim: Model sadece %0.4 krediyi temerrüt tahmin ediyor (gerçek oran 50x altı)
- Cozum 1: Baskin sinifi KUCULT (downsample)
- Yöntem: Baskin sınıfından rastgele N örneklem al (N = nadir sınıf sayısı)
- Sonuç: Dengeli eğitim seti; modelde iki sınıfın bilgi içeriği eşit
- Avantajı: Veri azaldığı için eğitim hızlanır
- Dezavantajı: Baskin sinifa ait bilgi KAYBEDİLİR - çok büyük veri setlerinde makul

Ağırlıklandırma - Veri Kaybetmeden Dengele

```
# Agirliklandirma: Nadir sinifa daha fazla agirlik ver
import numpy as np
from sklearn.linear_model import LogisticRegression

# Default = temerrut sinifi (nadir)
default_wt = 1 / np.mean(y == 'default')  # ~5.3
wt = [default_wt if yi == 'default' else 1 for yi in y]

# sample_weight ile agirlikli egitim
full_model = LogisticRegression(C=np.inf, max_iter=10000)
full_model.fit(X, y, sample_weight=wt)

# Tahmin oranlari artik dengeli
pred_rate = 100 * np.mean(full_model.predict(X) == 'default')
print(f'Temerrut tahmin orani: {pred_rate:.2f}%')
# Beklenen cikti: ~%58 (agirliksizda sadece %0.4)

# scikit-learn kisayolu: class_weight='balanced'
balanced_model = LogisticRegression(
    class_weight='balanced', C=np.inf, max_iter=10000)
balanced_model.fit(X, y)
```

SMOTE - Sentetik Oversampling

```
# SMOTE = Synthetic Minority Oversampling Technique
# Chawla et al. (2002) - nadir sinifa sentetik ornek uretir
# Fikir: Her nadir ornegin KOMSULARINA dayali yeni ornek olustur

from imblearn.over_sampling import SMOTE
from imblearn.pipeline import Pipeline

# SMOTE pipeline (yalnizca egitim verisinde kullanilir!)
smote_pipe = Pipeline([
    ('smote', SMOTE(sampling_strategy=0.8, random_state=42,
                    k_neighbors=5)),
    ('clf', LogisticRegression(max_iter=5000))
])

smote_pipe.fit(X_train, y_train)
y_pred = smote_pipe.predict(X_test)

# ONEMLI: SMOTE sadece egitim seti uzerinde uygulanir
# Test seti ORIJINAL dengesiz dagilimla kalmali (gercek dunyayi yansitmali)
```

Maliyete Dayalı Sınıflandırma

- Pratikte accuracy ve AUC kesim değeri seçmenin yolları
- Daha işletme-odaklı yaklaşım: False Positive ve False Negative maliyetlerini tahmin et
- Ornek (kredi): Beklenen getiri = P(Y=0) x R - P(Y=1) x C
- R = odenmis krediden kazanc, C = temerrudun maliyeti
- Krediye onay ver mi? Beklenen getirisi POZİTİF ise evet
- Avantajı: Modelden gelen olasılık + iş maliyeti birleşir -> optimal iş kararı
- Pazarlama: Her teklif maliyeti (postal, zaman) + beklenen gelir biliniyorsa model kesim değerini KENDİ'sine ekle

ÖZET

Bu Hafta Öğrendiklerimiz

- Sınıflandırma: İkili veya çok sınıflı kategorik Y değişkeni tahmini
- Naive Bayes: Koşullu bağımsızlık varsayımı, sıralama için iyi, biased olasılıklar
- LDA: Grup ayrımını maksimize eden doğrusal kombinasyon (Fisher 1936)
- Lojistik Regresyon: Logit dönüşümü + MLE, odds oranları ile yorumlanabilir
- Model degerlendirme: Confusion matrix, Precision, Recall, F1, AUC-ROC, Lift
- Accuracy dengesiz veride yanıltıcı -> daha nüanslı metrikler kullan
- Dengesiz veri: Undersampling, SMOTE, ağırlıklandırma, maliyete dayalı sınıflandırma

Kaynaklar ve Ileri Okuma

- Bruce, Bruce, Gedeck (2026). AI-Assisted Statistics for Data Scientists, 3. Baski, O'Reilly - Bolum 5
- Hosmer, Lemeshow, Sturdivant (2013). Applied Logistic Regression, 3rd ed., Wiley
- Hastie, Tibshirani, Friedman (2009). The Elements of Statistical Learning, 2nd ed., Springer
- James, Witten, Hastie, Tibshirani (2021/2023). An Introduction to Statistical Learning (R/Python)
- Chawla et al. (2002). SMOTE: Synthetic Minority Over-sampling Technique, J. of AI Research
- scikit-learn Dokumantasyonu: scikit-learn.org/stable/modules/linear_model.html#logistic-regression
- imbalanced-learn Dokumantasyonu: imbalanced-learn.org

Gelecek Hafta: Istatistiksel Makine Ogrenmesi

- K-En Yakin Komsu (KNN) - uzaklik tabanli siniflandirma
- Karar Agaclari - rekursif bolme ile tahmin
- Random Forest - bootstrap aggregation (bagging)
- XGBoost - gradient boosting
- Degisken onemi (feature importance) ve SHAP degerleri

Sorulariniz?